INVESTOR IN PEOPLE

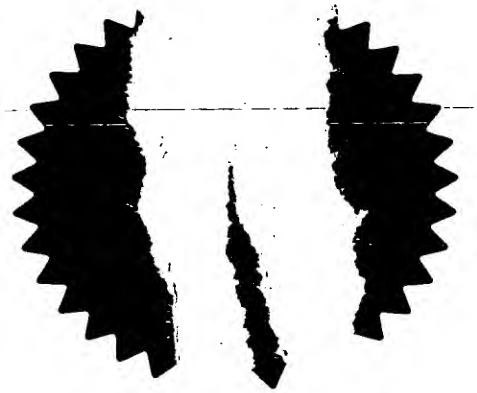
The Patent Office
Concept House
Cardiff Road
Newport
South Wales
NP10 8QQ

I, the undersigned, being an officer duly authorised in accordance with Section 74(1) and (4) of the Deregulation & Contracting Out Act 1994, to sign and issue certificates on behalf of the Comptroller-General, hereby certify that annexed hereto is a true copy of the documents as originally filed in connection with the patent application identified therein.

In accordance with the Patents (Companies Re-registration) Rules 1982, if a company named in this certificate and any accompanying documents has re-registered under the Companies Act 1980 with the same name as that with which it was registered immediately before re-registration save for the substitution as, or inclusion as, the last part of the name of the words "public limited company" or their equivalents in Welsh, references to the name of the company in this certificate and any accompanying documents shall be treated as references to the name with which it is so re-registered.

In accordance with the rules, the words "public limited company" may be replaced by p.l.c., plc, P.L.C. or PLC.

Re-registration under the Companies Act does not constitute a new legal entity but merely subjects the company to certain additional company law rules.



Signed

Dated 30 MAY 2001

Patents Form 1/77
Patents Act 1977
(Rule 16)

THE PATENT OFFICE
6 NOV 1998
LONDON

The Patent Office

6 NOV 1998 77

P01/7700 0.00-9824444.5

Request for grant of a patent

(See the notes on the back of this form. You can also get an explanatory leaflet from the Patent Office to help you fill in this form)

The Patent Office

Cardiff Road
Newport
Gwent NP9 1RH

1. Your reference — DCE/SW/28661 — 6 NOV 1998

2. Patent application number *(The Patent Office will fill in this part)* — **9824444.5**

3. Full name, address and postcode of the or of each applicant *(underline all surnames)*

THE MANCHESTER METROPOLITAN UNIVERSITY
BELLHOUSE BUILDING
LOWER ORMOND STREET
MANCHESTER M15 6BX

Patents ADP number *(if you know it)*

If the applicant is a corporate body, give the country/state of its incorporation — 7546922001

4. Title of the invention — MICRO-ORGANISM IDENTIFICATION

5. Name of your agent *(if you have one)* — fJ CLEVELAND

"Address for service" in the United Kingdom to which all correspondence should be sent *(including the postcode)*

40/43 CHANCERY LANE
LONDON WC2A 1JQ

Patents ADP number *(if you know it)* — 07368855001

6. If you are declaring priority from one or more earlier patent applications, give the country and the date of filing of the or of each of these earlier applications and *(if you know it)* the or each application number

Country	Priority application number *(if you know it)*	Date of filing *(day / month / year)*

7. If this application is divided or otherwise derived from an earlier UK application, give the number and the filing date of the earlier application

Number of earlier application	Date of filing *(day / month / year)*

8. Is a statement of inventorship and of right to grant of a patent required in support of this request? *(Answer 'Yes' if:*

a) any applicant named in part 3 is not an inventor, or
b) there is an inventor who is not named as an applicant, or
c) any named applicant is a corporate body.
See note (d))

YES

Patents Form 1/77

9. Enter the number of sheets for any of the following items you are filing with this form. Do not count copies of the same document

Continuation sheets of this form	
Description	9
Claim(s)	
Abstract	
Drawing(s)	3 FIGURES COMPRISING TOTAL OF 17 SHEETS

17

10. If you are also filing any of the following, state how many against each item.

Priority documents	
Translations of priority documents	
Statement of inventorship and right to grant of a patent *(Patents Form 7/77)*	
Request for preliminary examination and search *(Patents Form 9/77)*	
Request for substantive examination *(Patents Form 10/77)*	
Any other documents *(please specify)*	

11. I/We request the grant of a patent on the basis of this application.

Signature [signature] Date 6.11.98

12. Name and daytime telephone number of person to contact in the United Kingdom

MR DC EVANS - 0171 405 5875

Warning

After an application for a patent has been filed, the Comptroller of the Patent Office will consider whether publication or communication of the invention should be prohibited or restricted under Section 22 of the Patents Act 1977. You will be informed if it is necessary to prohibit or restrict your invention in this way. Furthermore, if you live in the United Kingdom, Section 23 of the Patents Act 1977 stops you from applying for a patent abroad without first getting written permission from the Patent Office unless an application has been filed at least 6 weeks beforehand in the United Kingdom for a patent for the same invention and either no direction prohibiting publication or communication has been given, or any such direction has been revoked.

Notes

a) *If you need help to fill in this form or you have any questions, please contact the Patent Office on 0645 500505.*

b) *Write your answers in capital letters using black ink or you may type them.*

c) *If there is not enough space for all the relevant details on any part of this form, please continue on a separate sheet of paper and write "see continuation sheet" in the relevant part(s). Any continuation sheet should be attached to this form.*

d) *If you have answered 'Yes' Patents Form 7/77 will need to be filed.*

e) *Once you have filled in the form you must remember to sign and date it.*

f) *For details of the fee and ways to pay please contact the Patent Office.*

Micro-orgarism identification

The present invention concerns the identification of microorganisms, and has particular reference to the differentiation between strains off bacteria. Our current pending patent application numbered WO 98/09314 describes and claims a method and apparatus for characterizing microorganisms using matrix assisted laser desorption ionisation time of flight mass spectrometry (MALDI-TOF-MS) spectral data for a range on known of microorganisms. The specification discloses that spectral data is included in a database and a sample of an unidentified microorganism is prepared and compared using suitable comparison means with the spectral data in the database.

MALDI-TOF-MS spectral data presents some difficulty in analysis since in the original analog spectral data, the intensities are not reproducible. In some spectra, the weak spectral peaks merge into the background "noise". Furthermore, the precision of the MALDI-TOF-MS machine is such that the mass position on each spectral peak is not exactly reproducible and a small element of "shift" for any given peak is likely to occur. This is particularly noticeable towards the high mass end of the spectrum.

Existing attempts to analyze the spectral data from MALDI-TOF-MS analysis have relied on the Jacquard method. According to this method, the spectral data is analyzed at a number of data points, typically at a number of data points greater than 16k. Each data point reports the presence or the absence of a peak at that particular point on the spectrum. The data point reports only the presence or the absence of a spectral peak and does not include any information what so ever concerning the intensity or relative intensity of any peak located at that position. The reported information from the data point is stored as an absolute number within the database. Using this technique there is no measure or relative intensity between the peaks and troughs or relative peaks within the spectrum being analyzed. Furthermore, because of the non-reproducibility of the spectral intensity, in some instances, significant but low intensity peaks will not be reported or considered. If the background noise level with in the system is relatively high, significant data may be lost due to it being simply discounted. Since the data set in any of one particular spectrum is very large and may be of the order of 16K. or 32K data points, significant and critical amount of characterizing information would simply be discounted with a result that critical comparison and analysis within the database cannot and could not take place.

In a small database, the time of calculation and

comparison is acceptable, but with a large database, a full comparison using the Jacquard method will take many days to complete. In order to reduce calculation times, it is necessary either to target only part of the spectral data or to discard some of the data from the total spectrum. In either case this results in a further degradation of potential accuracy, and positive identification or rejection is less likely to be obtained.

In the ideal analytical pattern recognition system, the system should report: --

(A) this example is of class "1" or

(B.) this example is from none of these classes or

(C) this example is too hard for me to consider.

The second category is called "outliers", while the third category is referred to as "rejects" or "doubt". Both categories of rejection have great importance in applications, particularly in medical diagnostic aids, where there is a clear need for certainty. A sample must either match, must be rejected out right, or must clearly be identified as the "doubtful"

For the foregoing, therefore, it will be seen that there is a need for an improved and more effective

diagnostic engine for use in the analysis of MALDI-TOF-MS spectral data.

According to one aspect of the present invention, therefore, there is provided a method off characterizing microorganisms which method comprises: --

providing a database of MALDI-TOF-MS spectral data for a range of known of microorganisms,

preparing a sample of unidentified microorganisms and obtaining the MALDI-TOF-MS spectral data there of

and comparing, using suitable comparison means, the spectral data so obtained with spectral data contained in the database, there by to identify a known microorganism having the same or similar spectral data,

characterized in that the comparison means comprises the steps of:-

defining a plurality of data points in the spectrum across the complete range of the spectral data, converting each data point to a vector spatial function, said function being characteristic of the position shape and relative intensity of the spectral data at that point

assembling the vector spatial functions for the spectrum in question as a cluster and then determining the kernel function in respect of the said cluster,

determining a radial base function for each kernel which is characteristic of all the information in that spectrum

and comparing that radial base function of the cluster kernel of the sample microorganism with the radial base function of the cluster kernel of all the other micro-organism spectra within the database.

In one aspect of the present invention, the spectral data may be normalized to provide an intensity function, which is a measure of the relative intensity of each spectral peak. In another embodiment, the normalization procedure may compare all the peak intensities as a proportion of the highest peak, which is rated at 1. All other peaks then have a value under one.

In another aspect of the present invention, the radial base function of the spectral data of media is applied across a neural network. The neural network may also be employed to analyze pattern distributions of radial base functions of the local kernel clusters using the Cover Theorem. In further aspect of the invention, the vector spatial functions of the

spectral data points may be displayed as a cluster in high dimensional space. The local kernel of each cluster of spectral data points in high dimensional space can be determined by a single set of searchable parameters. Thus instead of searching and comparing 16K. data points for each spectrum, all that is necessary is the comparison of the radial base functions of the local kernel clusters for each of the spectra within the database and compared it with the radial base functions of the local kernel cluster for the unknown sample. This has the effect of reducing the burden on the search engine while at the same time speeding up the search very considerably compared with methods hitherto employed or proposed.

The use of an artificial neural network to assist in optimization of the search data has the advantage that prior knowledge of models and associated careful network design is unnecessary. The use of a search engine in combination with MALDI-TOF-MS spectrum to make available high-performance mass spectral analysis tool, which may be operated by the non-specialist. The equipment required to perform the analysis is relatively inexpensive, and the search engine forming part of the invention enables rapid and easy searching of an extensive database of microorganisms.

The database in accordance with the present invention may comprise the radial base functions of the kernel

of each cluster of spectral data in hide dimensional space. In this way, none of the information relating to the spectrum is lost or discarded; and all of these included in the resulting radial base function of the cluster kernel and serve to determine the relative spatial position of the kernel in high dimensional space. This means that the spectral data may be recorded in digital form for ease of searching. The presence and availability of all the data points within the cluster for each spectrum permits the re-constitution of each spectrum from this information so that spectral data may be re-presented in graphic as well as digital or numeric form.

The invention also includes a database comprising the radial base functions of the known microorganisms for comparison with the organisms themselves.

Following is a description by way of example only of one method of carrying the invention into effect.

In the drawings: --

Figure 1 is a schematic representation of a neural network for use in the present invention.

Figure 2 is an algorithm for arriving at the radial base function for any particular spectrum.

Figure 3 is the detail of a program for use in the analytical process of the present invention.

The drawing of figure 1 is a schematic representation of a neural network, which can be adapted for use in the apparatus of the present invention. In this case, the radial base function of the kernel of the cluster of spectral data in respect of the sample is fed into the output neurone. This information is processed by a multitude of processors in the output layer and is presented at the output neurones. In the example shown in figure 1, a single output neurone is shown as the output layer. In accordance with the present invention, a multitude of output neurones would be provided, one in respect of each sample in the database available for comparison. The processed radial base function data is provided at each of the output neurones and is compared with the radial base function data for the sample with the corresponding function for each microorganism spectrum within the database. An exact match or a very close match will result in a clear identification of the sample microorganism.

Where there is no direct correspondence between the radial base function of the kernel of the data cluster for sample with corresponding radial base functions in the database, then a vector will be presented detailing the clusters in high dimensional space nearest to the radial base function of the

sample, which will give an indication of the degree of similarity or overlap between the unknown sample and the identified similar spectra within the database. This will enable the analyst to call up the graphic data relating to the particular "close matches" and to compare them visually.

It will be appreciated by the person skilled in the art that the radial base function of each cluster of spectral data in high dimensional space will be a result of all the features of each data point within the cluster and that the radial base function of kernel will be determined, spatially, by the individual values of the vector functions of each data point. Thus several similar microorganisms that are not identical may reside in the same proximate area of high dimensional space. The relative position of each kernel will be determined by the extent of the differences in their spectral details. If the microorganisms are of the same genus then the two kernels defined by the spectral clusters will substantially coincide, and the greater the extent of the overlap the greater the similarity of the microorganisms.

Figure 2 is an algorithm for determining the radial base functions of the cluster kernel for any given spectrum. Figure 3 is the detail of a computer program for performing the algorithm of figure 2.

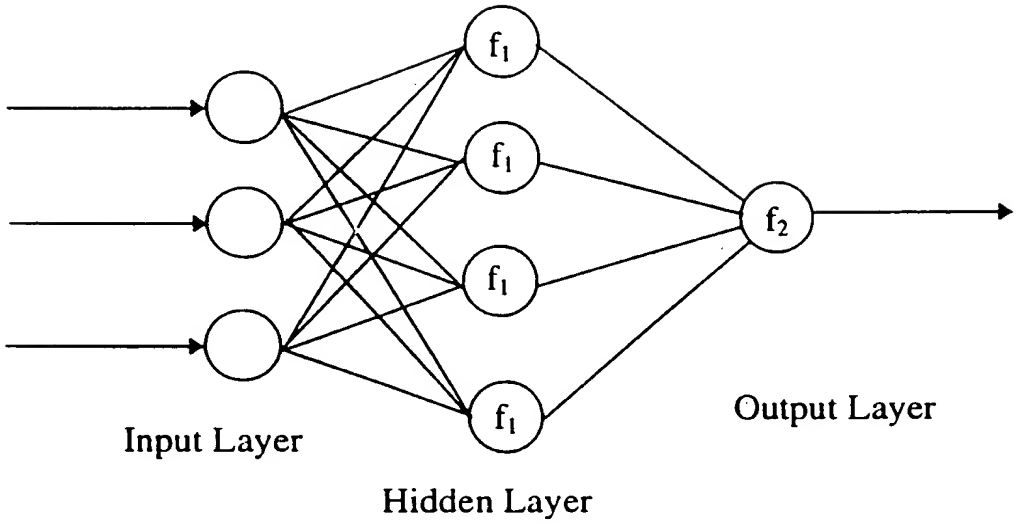


figure 1

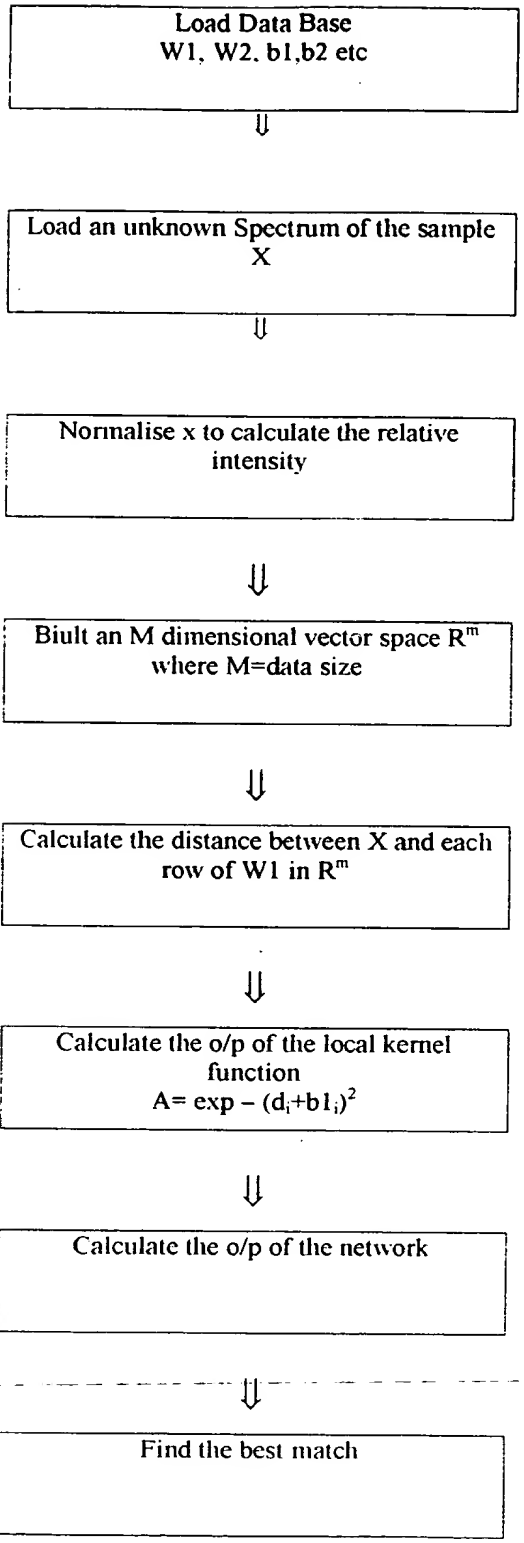

Load Data Base
W1, W2, b1,b2 etc
Load an unknown Spectrum of the sample
X
Normalise x to calculate the relative intensity
Biult an M dimensional vector space R^m where M=data size
Calculate the distance between X and each row of W1 in R^m
Calculate the o/p of the local kernel function
$A= \exp - (d_i+b1_i)^2$
Calculate the o/p of the network
Find the best match

FIGURE 2

FIGURE 3

Program Listing

```
#include <formatio.h>
#include <analysis.h>

#include <utility.h>
#include <ansi_c.h>
#include <userint.h>
#include "sample4.h"

#define NoData       125
#define DataSize     16370

static int panelHandle;

    static int status;
    static FILE *file_handle;
    static char pathname[MAX_PATHNAME_LEN];
    static char directory[MAX_PATHNAME_LEN];

int Load_Sample_OK=0;
int Load_DataBase_OK=0;

int err;
/*double mean_value;
  double datapoints[100];
*/

double P[DataSize];
double  P_index[DataSize];

double w1[NoData][DataSize];
double w2[NoData][NoData];
double b1[NoData];
double b2[NoData];

double a1[NoData];
double a2[NoData];

    double n[NoData];
    double x[DataSize];
    double y[DataSize];
    double y2[DataSize];
    double dist[NoData];


/****************************************************************************/

int main (int argc, char *argv[])
{
    if (InitCVIRTE (0, argv, 0) == 0)    /* Needed if linking in external compil
r; harmless otherwise */
        return -1;  /* out of memory */
    if ((panelHandle = LoadPanel (0, "sample4.uir", PANEL)) < 0)
        return -1;
```

```
    DisplayPanel (panelHandle);
    RunUserInterface ();
    return 0;
}

/*****************************************************************************/

int CVICALLBACK Shutdown (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    switch (event)
    {
        case EVENT_COMMIT:

            QuitUserInterface (0);

            break;

        case EVENT_RIGHT_DOUBLE_CLICK:

            break;
    }
    return 0;
}

/*****************************************************************************/

int CVICALLBACK AcquireData (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{

    double norm;

    int i=0;
    int j;
    double M,m;
    char buff[80];

    int err,s,q;

    switch (event) {

        case EVENT_COMMIT:

if(Load_DataBase_OK)
{
if(Load_Sample_OK)
{

        DeleteGraphPlot (panelHandle, PANEL_WAVEFORM_2, -1,
                                        VAL_IMMEDIATE_DRAW);

        SetCtrlVal (panelHandle, PANEL_ELEMENT, "Searching,Please Wait" );

        for (s=0; s<NoData; s++)
        {
        for (q=0; q<DataSize; q++)
        {
        x[q]=wl[s][q];
/*      x[q]=(x[q]-P[q])*(x[q]-P[q]);
        dist[s]=dist[s]+x[q];
*/
        }
```

```
        norm=0;
        Sub1D (x, P, DataSize, y);
        Mul1D (y, y, DataSize, y2);
        Sum1D (y2, DataSize, &norm);
        norm=sqrt(norm);
        dist[s]=norm;


/*      dist[s]=sqrt(dist[s]); */
        n[s]=dist[s]*b1[s];
        a1[s]=exp(-n[s]*n[s]);
/*
        for (q=0; q<NoData; q++)
        {
*/
        a1[s]=w2[s]*a1[s]+b2[s];
/*      n[i]=n[i]*b;
        a1[i] = exp (-n[i]*n[i]);
        datapoints[i] = rand()/32768.0;
        }
*/
    }

    MaxMin1D (a1, NoData, &M, &i, &m, &j);
    QScale1D (a1, NoData, a1, &M);
    j=i+1;

    PlotY (panelHandle, PANEL_WAVEFORM_2, a1, NoData, VAL_DOUBLE,
           VAL_THIN_LINE, VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_WHITE);


    switch (j)
        {

            case 0:

                Fmt (buff, "%s (%d)","No Sample was selected" , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 1:

                Fmt (buff, "%s (%d)","Acina, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 2:

                Fmt (buff, "%s (%d)","alcal, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 3:

                Fmt (buff, "%s (%d)","baccer, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 4:
```

```
        Fmt (buff, "%s (%d)","baccer2, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 5:

        Fmt (buff, "%s (%d)","bacmy, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 6:

        Fmt (buff, "%s (%d)","bacsub, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 7:

        Fmt (buff, "%s (%d)","citd, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 8:

        Fmt (buff, "%s (%d)","citd2, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 9:

        Fmt (buff, "%s (%d)","citf, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 10:

        Fmt (buff, "%s (%d)","citf2, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 11:

        Fmt (buff, "%s (%d)","citf3, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 12:

        Fmt (buff, "%s (%d)","ecoli1, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 13:

        Fmt (buff, "%s (%d)","ecoli2, Elment No." , i);
```

```
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 14:

        Fmt (buff, "%s (%d)","ecoli3, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 15:

        Fmt (buff, "%s (%d)","ecoli4, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 16:

        Fmt (buff, "%s (%d)","ecoli26, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 17:

        Fmt (buff, "%s (%d)","ecoli27, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 18:

        Fmt (buff, "%s (%d)","ecoli28, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 19:

        Fmt (buff, "%s (%d)","ecoli29, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 20:

        Fmt (buff, "%s (%d)","ecoli30, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 21:

        Fmt (buff, "%s (%d)","ecoli31, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 22:

        Fmt (buff, "%s (%d)","ecoli32, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);
```

```
        break;

    case 23:

        Fmt (buff, "%s (%d)","ecoli33, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 24:

        Fmt (buff, "%s (%d)","ecoli34, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 25:

        Fmt (buff, "%s (%d)","ecoli35, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 26:

        Fmt (buff, "%s (%d)","ent, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 27:

        Fmt (buff, "%s (%d)","entc, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 28:

        Fmt (buff, "%s (%d)","entd, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 29:

        Fmt (buff, "%s (%d)","entf, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 30:

        Fmt (buff, "%s (%d)","entf2, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 31:

        Fmt (buff, "%s (%d)","entf3, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);
```

```
        break;

    case 32:

        Fmt (buff, "%s (%d)","entf4, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 33:

        Fmt (buff, "%s (%d)","entf5, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 34:

        Fmt (buff, "%s (%d)","entf6, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 35:

        Fmt (buff, "%s (%d)","entf7, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 36:

        Fmt (buff, "%s (%d)","entvre, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 37:

        Fmt (buff, "%s (%d)","gon1, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 38:

        Fmt (buff, "%s (%d)","gon2, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 39:

        Fmt (buff, "%s (%d)","hafalv, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 40:

        Fmt (buff, "%s (%d)","keba, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;
```

```
case 41:

    Fmt (buff, "%s (%d)","listg, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 42:

    Fmt (buff, "%s (%d)","listi, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 43:

    Fmt (buff, "%s (%d)","listm, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 44:

    Fmt (buff, "%s (%d)","listm3, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 45:

    Fmt (buff, "%s (%d)","listm4, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 46:

    Fmt (buff, "%s (%d)","listmu, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 47:

    Fmt (buff, "%s (%d)","mening1, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 48:

    Fmt (buff, "%s (%d)","mening2, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 49:

    Fmt (buff, "%s (%d)","nelong, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;
```

```
case 50:

    Fmt (buff, "%s (%d)","nflav1, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 51:

    Fmt (buff, "%s (%d)","nflav2, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 52:

    Fmt (buff, "%s (%d)","nsicca, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 53:

    Fmt (buff, "%s (%d)","pro1, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 54:

    Fmt (buff, "%s (%d)","pro2, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 55:

    Fmt (buff, "%s (%d)","pro3, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 56:

    Fmt (buff, "%s (%d)","prov1, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 57:

    Fmt (buff, "%s (%d)","pseu1, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 58:

    Fmt (buff, "%s (%d)","pseua2, Elment No." , i);
    SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

    break;

case 59:
```

```
        Fmt (buff, "%s (%d)","pseua3, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 60:

        Fmt (buff, "%s (%d)","sal, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 61:

        Fmt (buff, "%s (%d)","salg1, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 62:

        Fmt (buff, "%s (%d)","salg10, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 63:

        Fmt (buff, "%s (%d)","salg78, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 64:

        Fmt (buff, "%s (%d)","salt2, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 65:

        Fmt (buff, "%s (%d)","saltyp, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 66:

        Fmt (buff, "%s (%d)","saltyp1, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 67:

        Fmt (buff, "%s (%d)","serrat, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 68:
```

```
        Fmt (buff, "%s (%d)","shig, Element No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 69:

        Fmt (buff, "%s (%d)","stapha1, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 70:

        Fmt (buff, "%s (%d)","stapha2 Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 71:

        Fmt (buff, "%s (%d)","stapha3, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 72:

        Fmt (buff, "%s (%d)","strep1, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 73:

        Fmt (buff, "%s (%d)","strep2, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 74:

        Fmt (buff, "%s (%d)","beco16k1, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 75:

        Fmt (buff, "%s (%d)","beco16k2, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 76:

        Fmt (buff, "%s (%d)","beco16k3, Elment No." , i);
        SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

        break;

    case 77:

        Fmt (buff, "%s (%d)","beco16k4, Elment No." , i);
```

```
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 78:

                Fmt (buff, "%s (%d)","psta16k1, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 79:

                Fmt (buff, "%s (%d)","psta16k2, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            case 80:

                Fmt (buff, "%s (%d)","psta16k3, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

            default:

                Fmt (buff, "%s (%d)","Unknown, Elment No." , i);
                SetCtrlVal (panelHandle, PANEL_ELEMENT, buff);

                break;

        }

        YGraphPopup ("Result of Search Engine for the Selected Sample",
                     a1, NoData, VAL_DOUBLE);

}

    else
    {

    MessagePopup ("SAMPLE", "Please Load a Sample First");

    }
}

    else
    {

    MessagePopup ("DATA BASE", "Please Load the Data Base First");

    }

        break;

        case EVENT_RIGHT_DOUBLE_CLICK:

            break;
    }
    return 0;
}

/*****************************************************************/
```

```
int CVICALLBACK LoadDataBase (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{

    int s,q, num;

    switch (event)

    {
        case EVENT_COMMIT:

SetCtrlVal (panelHandle, PANEL_ELEMENT, "loading,Please Waite" );

GetProjectDir (pathname);
status = FileSelectPopup (directory, "w100.mat", "*.Mat(Mat file)",
                          "Data File", VAL_LOAD_BUTTON, 0, 0, 1, 0,
                          pathname);

if (status != VAL_NO_FILE_SELECTED)
{
   file_handle = fopen (pathname, "r");
   for (s=0;s<NoData;s++)
   {
    for (q=0; q<DataSize; q++)
    {
    num = fscanf(file_handle,"%lf",&w1[s][q]);
    }
   }
   fclose(file_handle);
    Load_DataBase_OK=1;

}
else
{

Load_DataBase_OK=0;

}


GetProjectDir (pathname);
status = FileSelectPopup (directory, "b100.mat", "*.Mat(Mat file)",
                          "Data File", VAL_LOAD_BUTTON, 0, 0, 1, 0,
                          pathname);
if (status != VAL_NO_FILE_SELECTED)
{
   file_handle = fopen (pathname, "r");
   for (s=0;s<NoData;s++)
   {
    num = fscanf(file_handle,"%lf",&b1[s]);
   }
   fclose(file_handle);
}
else
{

Load_DataBase_OK=0;

}

SetCtrlVal (panelHandle, PANEL_ELEMENT, "load a sample or search" );

            break;
```

```
        case EVENT_RIGHT_DOUBLE_CLICK:

            break;
    }
    return 0;
}
/*****************************************************************/

int CVICALLBACK Load_Sample (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    int i, num,pmax_index,pmin_index;
    double  pmax,pmin;
    switch (event)
    {


        case EVENT_COMMIT:

DeleteGraphPlot (panelHandle, PANEL_WAVEFORM, -1, VAL_IMMEDIATE_DRAW);

SetCtrlVal (panelHandle, PANEL_ELEMENT, "Loading,Please Waite" );

GetProjectDir (pathname);
status = FileSelectPopup (directory, "*.16k", "*.dat(data.dat)",
                          "Data File", VAL_LOAD_BUTTON, 0, 0, 1, 0,
                          pathname);
if (status != VAL_NO_FILE_SELECTED)
{
   file_handle = fopen (pathname, "r");

   for (i=0;i<DataSize;i++)

   {

   num = fscanf (file_handle, "%lf%lf\n", &P_index[i],&P[i]);

   }

    fclose(file_handle);

    Load_Sample_OK=1;

    SetCtrlVal (panelHandle, PANEL_ELEMENT, "Unknown Sample");

    MaxMin1D (P, DataSize, &pmax, &pmax_index, &pmin, &pmin_index);

    QScale1D (P, DataSize, P_index, &pmax);

    for (i=0; i<DataSize;i++) P[i]=3000*P_index[i];

    PlotY (panelHandle, PANEL_WAVEFORM, P_index, DataSize, VAL_DOUBLE,
           VAL_THIN_LINE, VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_YELLOW);

/*  PlotY (panelHandle, PANEL_WAVEFORM, P, DataSize, VAL_DOUBLE,
           VAL_THIN_LINE, VAL_EMPTY_SQUARE, VAL_SOLID, 1, VAL_YELLOW);

/*  SetAxisScalingMode (panelHandle, PANEL_WAVEFORM, VAL_XAXIS,
                        VAL_MANUAL, 500, 10000);
/*  SetAxisRange (panelHandle, PANEL_WAVEFORM, VAL_NO_CHANGE, 500, 10000,
                  VAL_AUTOSCALE, 0.0, 1.0);
*/
```

```
}
else
{
Lo[redacted]_Sample_OK=0;
}
            break;
        case EVENT_RIGHT_DOUBLE_CLICK:
            break;
    }
    return 0;
}
/*****************************************************************************/
```